

CLAIMS

What is claimed is:

1. A method of processing an instruction in a processor,
said method comprising:

executing a first instruction and storing data
associated with executing said first instruction;

if said first instruction was executed out-of-
order, determining whether the stored data should have
been used in execution of a preceding second instruction
executed after said first instruction; and

if the stored data should have been used to
execute the second instruction, using the stored data to
perform an operation indicated by said second instruction
without re-executing said second instruction.

2. A processor, comprising:

means for executing a first instruction and storing data associated with executing said first instruction;

means for determining, if said first instruction was executed out-of-order, whether the stored data should have been used in execution of a preceding second instruction executed after said first instruction; and

means for using the stored data to perform an operation indicated by said second instruction without re-executing said second instruction, if the stored data should have been used to execute the second instruction.

3. A processor, comprising:

```
a register set;
```

at least one execution unit that executes load instructions to transfer data into said register set;

a load queue containing at least one entry,
wherein said entry stores load data retrieved by a first
load instruction; and

queue management logic that, responsive to execution of a second load instruction, detects by reference to said load queue whether a data hazard exists, and if so, outputs said load data retrieved by said first load instruction from said entry to said register set in accordance with said second load instruction.

1 4. The processor of Claim 3, wherein said entry stores
2 a target address of said first load instruction and has a
3 hazard flag indicative of a possible data hazard, wherein
4 said queue management logic detects that a data hazard
5 exists if said second load instruction precedes said
6 first instruction in program order and a target address
7 of said second load instruction matches said target
8 address stored in said entry and said hazard flag is set.

1 5. The processor of Claim 3, wherein said queue
2 management logic sets said hazard flag at least in
3 response to local store operation specifying said target
4 address.
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

1 6. The processor of Claim 3, said register set
2 comprising a general purpose register set.

1 7. The processor of Claim 3, wherein said queue
2 management logic outputs said load data to a register in
3 said register set that is specified by said second load
4 instruction.

1 8. The processor of Claim 3, wherein said queue
2 management logic, responsive to detection of a data
3 hazard, initiates reexecution of at least said first load
4 instruction.

1 9. The processor of Claim 3, wherein said queue
2 management logic allocates a respective entry within said
3 load queue to each load instruction upon dispatch and,
4 upon completion of said each load instruction,
5 deallocates said respective entry.

002290"4249550

1 10. A data processing system, comprising:

2 an interconnect fabric;

3 a memory coupled to said interconnect fabric;

4 a register set;

5 at least one execution unit that executes load
6 instructions to transfer data from said memory into said
7 register set;

8 a load queue containing at least one entry,
9 wherein said entry stores load data retrieved by a first
10 load instruction; and

11 queue management logic that, responsive to
12 execution of a second load instruction, detects by
13 reference to said load queue whether a data hazard
14 exists, and if so, outputs said load data retrieved by
15 said first load instruction from said entry to said
16 register set in accordance with said second load
17 instruction.

1 11. The data processing system of Claim 10, wherein said
2 entry stores a target address of said first load
3 instruction and has a hazard flag indicative of a
4 possible data hazard, wherein said queue management logic
5 detects that a data hazard exists if said second load
6 instruction precedes said first instruction in program
7 order and a target address of said second load
8 instruction matches said target address stored in said
9 entry and said hazard flag is set.

12. The data processing system of Claim 11, wherein said
queue management logic sets said hazard flag at least in
response to local store operation specifying said target
address.

13. The data processing system of Claim 12, wherein said
at least one execution unit, said register set and said
load queue comprise a first processor and said data
processing system includes a second processor, wherein
said queue management logic also sets said hazard flag in
response to said second processor issuing an exclusive
access operation specifying said target address on said
interconnect fabric.

14. The data processing system of Claim 10, said
register set comprising a general purpose register set.

15. The data processing system of Claim 10, wherein said queue management logic outputs said load data to a register in said register set that is specified by said second load instruction.

16. The data processing system of Claim 10, wherein said queue management logic, responsive to detection of a data hazard, initiates reexecution of at least said first load instruction.

17. The data processing system of Claim 10, wherein said queue management logic allocates a respective entry within said load queue to each load instruction upon dispatch and, upon completion of said each load instruction, deallocates said respective entry.

18. A method of executing load instructions out-of-order in a processor having a register set and a load queue, said method comprising:

storing, in an entry in said load queue, load data retrieved from memory in response to executing a first load instruction;

in response to execution of a second load instruction, detecting by reference to said load queue whether a data hazard exists; and

in response to detection of a data hazard,
outputting said load data retrieved by said first load
instruction from said entry to said register set in
accordance with said second load instruction.

1 19. The method of Claim 18, wherein said entry stores a
2 target address of said first load instruction and has a
3 hazard flag indicative of a possible data hazard, wherein
4 detecting that a data hazard exists comprises determining
5 if said second load instruction precedes said first
6 instruction in program order and a target address of said
7 second load instruction matches said target address
8 stored in said entry and said hazard flag is set.

1 20. The method of Claim 19, and further comprising
2 setting said hazard flag at least in response to a local
3 store operation specifying said target address.

1 21. The method of Claim 19, wherein outputting said load
2 data comprises outputting said load data to a register in
3 said register set that is specified by said second load
4 instruction.

1 22. The method of Claim 19, and further comprising:

2 in response to detection of a data hazard,
3 initiating reexecution of at least said first load
4 instruction.

1 23. The method of Claim 19, and further comprising
2 allocating a respective entry within said load queue to
3 each load instruction upon dispatch and, upon completion
4 of said each load instruction, deallocating said
5 respective entry.